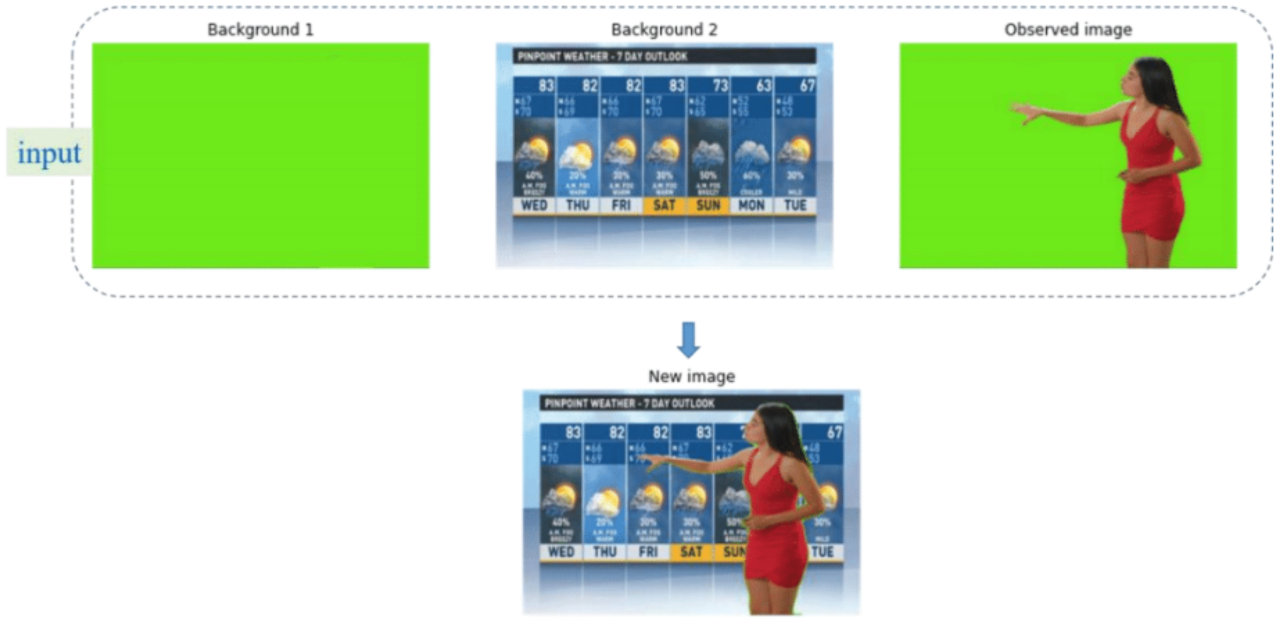
VGU – CSE2022

Exercise 5 - Project

October 18, 2022

Write a function to performed background subtraction, as illustrated following



- **Library to read an image:**
  https://github.com/nothings/stb/blob/master/stb_image.h
- **Library to write an image:**
  https://github.com/nothings/stb/blob/master/stb_image_write.h
- **Test image:**
  https://raw.githubusercontent.com/neko941/BALOS/main/images/98239648_p0.png
- **Directory structure:**

```
├─ headers
│   ├─ stb_image.h
│   └─ stb_image_write.h
├─ images
│   └─ 98239648_p0.png
└─ main.c
```

- **Example codes:**

```c
#include <stdio.h>

#define STB_IMAGE_IMPLEMENTATION
#include "./headers/stb_image.h"
#define STB_IMAGE_WRITE_IMPLEMENTATION
#include "./headers/stb_image_write.h"

/**
 * Delete a quarter of the image
 * @param[in] image the input image
 * @param[in] width the width of the image
 * @param[in] height the height of the image
 * @param[in] channel the channel of the image
 */
unsigned char mask_image(unsigned char *image, int width, int height, int channel
    )
{
    for (int i = 0; i < height / 2; i++)
    {
        for (int j = 0; j < width / 2; j++)
        {
            for (int k = 0; k < channel; k++)
            {
                image[i * width * channel + j * channel + k] = 0;
            }
        }
    }
}

int main()
{
    // declare variables
    int width, height, channel;
    char path_img[] = "./images/98239648_p0.png";
    char save_path[] = "./images/98239648_p0-New.png";

    // read image data
    unsigned char *image = stbi_load(path_img, &width, &height, &channel, 0);
    if (image == NULL)
    {
        printf("\nError in loading the image\n");
        exit(1);
    }
    printf("Width = %d\nHeight = %d\nChannel = %d\n", width, height, channel);

    // fill image with black pixels
    mask_image(image, width, height, channel);

    // save image
    stbi_write_png(save_path, width, height, channel, image, width * channel);
    printf("New image saved to %s\n", save_path);
}

```

Code Listing 1: Delete a quarter of the image

```c
#include <math.h>
#include <stdio.h>

#define STB_IMAGE_IMPLEMENTATION
#include "./headers/stb_image.h"
#define STB_IMAGE_WRITE_IMPLEMENTATION
#include "./headers/stb_image_write.h"

/**
 * Create a new 1-dimensional array with the given size
 * @param[in] _size the size of the array
 * @param[out] _ empty 1-dimensional array filled with 0
 */
unsigned char *uc_arrayNew_1d(int _size)
{
    return (unsigned char *)calloc(_size, sizeof(unsigned char));
}

/**
 * Rotate image with arbitrary angle
 * @param[in] image image to be rotated
 * @param[in] width width of image
 * @param[in] height height of image
 * @param[in] channel channel of image
 * @param[in] degree angle of rotation
 * @param[out] _ rotated image
 */
unsigned char * image_rotation(unsigned char *image, int width, int height, int
    channel, int degrees)
{
    unsigned char *tary = uc_arrayNew_1d(width * height * channel);
    float radians = degrees * M_PI / 180.0;
    float xcenter = (float)(width) / 2.0;
    float ycenter = (float)(height) / 2.0;
    for (int i = 0; i < height; ++i)
    {
        for (int j = 0; j < width; ++j)
        {
            for (int k = 0; k < channel; k++)
            {
                int rorig = ycenter + ((float)(i)-ycenter) * cos(-radians) - ((
    float)(j)-xcenter) * sin(-radians);
                int corig = xcenter + ((float)(i)-ycenter) * sin(-radians) + ((
    float)(j)-xcenter) * cos(-radians);
                if (rorig >= 0 && rorig < height && corig >= 0 && corig < width)
                {
                    tary[i * width * channel + j * channel + k] = image[rorig *
    width * channel + corig * channel + k];
                }
            }
        }
    }
    return tary;
}

int main()
{
    // declare variables
    int width, height, channel;
    char path_img[] = "./images/98239648_p0.png";
    char save_path_rotate[] = "./images/98239648_p0-Rotated.png";

    // read image data
    unsigned char *image = stbi_load(path_img, &width, &height, &channel, 0);
    if (image == NULL)
    {
        printf("\nError in loading the image\n");
        exit(1);
    }
    printf("Width = %d\nHeight = %d\nChannel = %d\n", width, height, channel);

    // roate the image
    unsigned char *rimage = image_rotation(image, width, height, channel, 230);

    // save image
    stbi_write_png(save_path_rotate, width, height, channel, rimage, width *
   channel);
    printf("New image saved to %s\n", save_path_rotate);
}

```

Code Listing 2: Rotate the image with an arbitrary angle